Digital Loggers

- About
- Products
- Support
- Quotes
- Applications

- Power Control
- Pro Switch
- Rack Mount PDU
- New 90-240V PDU
- 3-Phase PDU
- DC Power Switch
- DIN Power Relay
- Gigabit Midspan PoE
- 24v PoE Injector

- Industrial IoT
- IoT WiFi PLC
- Atomic Pi

- **IoT Power Relay**
- **RS-232 Switch**

- **USB Loggers**
- **16-Channel DIY**
- **T1/PRI Logger**
- **Personal Logger**
- **Accessories**

- **Logging Systems**
- **Win 10 Appliance**
- **NG911 & Airband**
- **Military Radios**

Only the WiFi capable, Atheros-based power controllers support the REST API.

Download the REST API Reference here.

These examples are using cURL. ***Some curl versions may not authenticate properly when using digest authentication.**
This version of curl is tested and works well in Windows.

To use the REST API, enable the "Allow REST-style API" on the External APIs page of the power controller

REST API Power Switch Control Examples

**Only the admin user can run scripts.
Non-admin users must be granted access in the External APIs settings, but then can only toggle outlets.

POWER CONTROLLER DEFINITIONS
Persistent state - The outlet state will revert to the persistent state after a power cycle or reboot.
Set persistent state: curl --digest -u admin:1234 -X PUT -H "X-CSRF: x" --data "value=true"
"http://192.168.0.100/restapi/relay/outlets/2/state/"
Get persistent state: curl -u admin:1234 -k -H "Accept:application/json" --digest
https://192.168.0.100/restapi/relay/outlets/2/state/
*Geting the persistent state may not reflect the physical state, if the transient state has been set.

Transient state - Temporary outlet state. The outlet may not return to the set state, but will revert to the persistent state after a power cycle or reboot.
Set transient state: curl --digest -u admin:1234 -X PUT -H "X-CSRF: x" --data "value=true"
"http://192.168.0.100/restapi/relay/outlets/2/transient_state/"
Get transient state: curl -u admin:1234 -k -H "Accept:application/json" --digest
https://192.168.0.100/restapi/relay/outlets/2/transient_state/
*Getting the transient state is "usually" the pysical state, but if there is a delay, e.g. cycle(), the physical state may not have been set yet.

Physical state - Get current physical outlet state of the outlet/relay.
Physical state: curl -u admin:1234 -k -H "Accept:application/json" --digest
https://192.168.0.100/restapi/relay/outlets/2/physical_state/
*The physical state cannot be set by the user. Users set the Transient state or Persistent state. The unit will change the physical state.

Note: Setting the "state" or "transient state" to "on" may not happen immediatey, as the on sequence delays and cycle delays will be honored.

Look at the API - level 1
curl -u admin:1234 -H "Range: dli-depth=1" -H "Accept: application/json" --digest "http://192.168.0.100/restapi/"

Outlet/relay control examples. Relays are zero based (0-7).

Switch relay 3 on. (true=on false=off)
curl --digest -u admin:1234 -X PUT -H "X-CSRF: x" --data "value=true" "http://192.168.0.100/restapi/relay/outlets/2/state/"

Turn all relays on.
curl --digest -u admin:1234 -X PUT -H "X-CSRF: x" --data "value=true"
"http://192.168.0.100/restapi/relay/outlets/all;/state/"

Turn relays 1 and 5 on.
curl -u admin:1234 -X PUT -H "X-CSRF: x" --data "value=true" --digest
"http://192.168.0.100/restapi/relay/outlets/=0,4/state/"

Get the physical status of relay/outlet 3
curl -k -u admin:1234 -H "Accept:application/json" --digest https://192.168.0.100/restapi/relay/outlets/2/physical_state/

Get the physical status of all relays
curl -k -u admin:1234 -H "Accept:application/json" --digest "https://192.168.0.100/restapi/relay/outlets/all;/physical_state/"

Get the names of all relays
curl -u admin:1234 -H "Accept:application/json" --digest "http://192.168.0.100/restapi/relay/outlets/all;/name/"

Cycle relay/outlet 1
curl -u admin:1234 -X POST -H "X-CSRF: x" --digest "http://192.168.0.100/restapi/relay/outlets/0/cycle/"

Running scripts

Run a script (flash_a_light)
curl -u admin:1234 --digest -H "X-CSRF: x" -H "Content-Type: application/json" --data "
[{\"user_function\":\"flash_a_light\"}]" http://192.168.0.100/restapi/script/start/

Run a script passing arguments (cycle_an_outlet(outlet number, interval)) and HTTPS
curl -k -u admin:4321 -H "X-CSRF: x" -H "Content-Type: application/json" --digest --data-binary "
[{\"user_function\":\"cycle_an_outlet\",\"source\":\"cycle_an_outlet(5,10)\"}]" "https://192.168.0.100/restapi/script/start/"

Show the running threads (scripts)
curl -u admin:1234 -X GET -H "Accept: application/json" --digest http://192.168.0.100/restapi/script/threads/

Stop a running thread (by Thread ID)
-u admin:1234 --digest -H "X-CSRF: x" -H "Content-Type: application/json" --data "[\"20\"]"
http://192.168.0.100/restapi/script/stop/

Stop all running threads (scripts)
curl -u admin:1234 --digest -H "X-CSRF: x" -H "Content-Type: application/json" --data "[\"all\"]"
http://192.168.0.100/restapi/script/stop/

Making configuration changes.

Enable SSH
curl curl -u admin:1234 -X PUT -H "X-CSRF: x" --data "value=true" --digest
"http://192.168.0.100/restapi/config/ssh_enabled/"

Add a user, allowing access to outlets 5, 6, 7 and 8
curl --digest -u admin:1234 -H "Content-type: application/json" -H "X-Requested-With: XMLHttpRequest" --data-binary "
{\"password\":\"aPassword\",\"is_admin\":false,\"name\":\"aUsername\",\"is_allowed\":true,\"outlet_access\":
[false,false,false,false,true,true,true,true]}" "http://192.168.0.100/restapi/auth/users/"

Delete a user
curl --digest -u admin:1234 -X DELETE -H "X-Requested-With: XMLHttpRequest"
"http://192.168.0.100/restapi/auth/users/1/"

Getting info from meters (when equipped). Buses and sensors are zero indexed.

Get the EPCR Bus 2 Voltage (zero indexed)
curl -u admin:1234 -H "Accept:application/json" http://192.168.0.100/restapi/meter/values/buses.1.voltage/value/

Get the EPCR Bus 1 (zero indexed) Current. This will be a small number if zero (e.g. 1.4e-45), so you'll need to round it.
curl -u admin:1234 -H "Accept:application/json" http://192.168.0.100/restapi/meter/values/buses.0.current/value/

Get the EPCR Bus 1 Total Energy Useage in Joules. (1 kWh = Joules * 0.00000027778)
curl -u admin:1234 -H "Accept:application/json" http://192.168.0.100/restapi/meter/values/buses.0.total_energy/value/

EPCR internal temperature (degrees kelvin) Celsius=kelvin - 273.16 Farenheight=9 / 5 * (kelvin - 273.16) + 32
curl -u admin:1234 -H "Accept:application/json" http://192.168.0.100/restapi/meter/values/environment.temperature/value/

EPCR/DIN4 sensor temperature (degrees kelvin)
curl -u admin:1234 -H "Accept:application/json" http://192.168.0.100/restapi/meter/values/sensors.0.temperature/value/

EPCR/DIN4 sensor humidity
curl -u admin:1234 -H "Accept:application/json"
http://192.168.0.100/restapi/meter/values/sensors.0.relative_humidity/value/

Have a smart way to use your power switch? Share it with us. We'll acknowledge your contribution. Learn more about scripting here or AutoPing here.